CICLO FORMATIVO DE GRADO SUPERIOR DESARROLLO DE APLICACIONES WEB

CURSO 2023/2024

Módulo : Desarrollo Web en Entorno Cliente
Práctica : Proyecto

Victor Stala.

Memoria

1. Ventajas y desventajas de utilizar mecanismos de comunicación asíncrona entre cliente y servidor Web:

Ventajas:

- Respuesta más rápida: Al ser asíncrona, la comunicación permite que el cliente no tenga que esperar a que se complete una solicitud antes de realizar otra acción, lo que puede mejorar significativamente la experiencia del usuario.
- Eficiencia en el uso de recursos: La comunicación asíncrona permite manejar múltiples solicitudes sin necesidad de mantener conexiones abiertas durante largos períodos, lo que puede reducir la carga del servidor y mejorar la escalabilidad.
- Mejora la experiencia del usuario: Al permitir actualizaciones de contenido sin recargar toda la página, la comunicación asíncrona puede crear interfaces más dinámicas y fluidas.
- Manejo de errores mejorado: Al no bloquear el flujo de ejecución, los errores en una solicitud asíncrona pueden manejarse de manera más efectiva sin interrumpir otras operaciones.

Desventajas:

- Complejidad de implementación: El manejo de la asincronía puede ser más complicado que las operaciones sincrónicas, lo que puede aumentar la complejidad del código y la posibilidad de errores.
- Posibles problemas de rendimiento: Si no se gestiona correctamente, el uso excesivo de comunicación asíncrona puede llevar a problemas de rendimiento, como la congestión de red o la sobrecarga del servidor.
- Mayor carga de procesamiento en el cliente: La comunicación asíncrona puede requerir más recursos del cliente, como memoria y capacidad de procesamiento, especialmente si se utilizan tecnologías complejas como WebSockets o Web Workers.
- Posibles problemas de seguridad: La asincronía puede introducir desafíos adicionales en términos de seguridad, como la posibilidad de ataques de denegación de servicio (DoS) si no se gestionan adecuadamente las solicitudes asíncronas.

2. Explicación del mecanismo de la comunicación asíncrona:

La comunicación asíncrona es un enfoque en el cual las solicitudes y respuestas entre el cliente y el servidor no se manejan de manera sincrónica, es decir, no se espera a que una solicitud se complete antes de enviar otra. En lugar de eso, el cliente puede enviar múltiples solicitudes y recibir respuestas en cualquier orden, sin bloquear el flujo de ejecución.

Este enfoque se logra típicamente utilizando tecnologías como XMLHttpRequest (XHR), Fetch API, WebSockets, o Server-Sent Events (SSE). En lugar de esperar a que se complete una solicitud antes de enviar otra, el cliente puede seguir ejecutando su código mientras espera las respuestas del servidor. Esto permite una experiencia de usuario más fluida y receptiva, ya que el cliente puede actualizar partes específicas de la página sin tener que recargarla por completo.

3. Análisis de las propiedades y métodos de los objetos implicados en la comunicación asíncrona (XMLHttpRequest):

XMLHttpRequest (XHR) es un objeto que permite realizar solicitudes HTTP asíncronas desde el navegador web. Algunas de sus propiedades y métodos más importantes incluyen:

Propiedades:

- readyState: Indica el estado de la solicitud (0: sin inicializar, 1: cargando, 2: cargado, 3: interactiva, 4: completado).
- status: Código de estado HTTP devuelto por el servidor.
- responseText / responseXML: Contienen los datos devueltos por el servidor en forma de texto o XML, respectivamente.
- Métodos:
- open(method, url, async): Inicializa una solicitud. El parámetro method especifica el método HTTP (GET, POST, etc.), url es la URL del servidor y async indica si la solicitud es asíncrona o no.
- send(data): Envía la solicitud al servidor. El parámetro data es opcional y se utiliza para enviar datos con la solicitud (por ejemplo, en el caso de solicitudes POST).
- abort(): Cancela la solicitud actual.
- setRequestHeader(header, value): Establece el valor de una cabecera HTTP personalizada para la solicitud.

4. Clasificación y análisis de las librerías actuales que facilitan la incorporación de las tecnologías de actualización dinámica a la programación de páginas Web:

Existen diversas librerías y frameworks que facilitan la implementación de actualizaciones dinámicas en páginas web. Algunas de las más populares son:

- React: React es una biblioteca de JavaScript desarrollada por Facebook que se utiliza principalmente para construir interfaces de usuario interactivas y dinámicas. Su enfoque en el manejo eficiente del estado y el renderizado virtual lo hace ideal para crear aplicaciones web que requieren actualizaciones dinámicas.
- Vue.js: Vue.js es otro framework de JavaScript que se utiliza para construir interfaces de usuario interactivas y dinámicas. Es conocido por su curva de aprendizaje suave y su flexibilidad, lo que lo hace popular entre los desarrolladores que desean agregar funcionalidades dinámicas a sus páginas web de manera rápida y sencilla.
- Angular: Angular es un framework desarrollado por Google para construir aplicaciones web de una sola página (SPA). Ofrece características avanzadas para la creación de aplicaciones web complejas, incluyendo la capacidad de actualizar dinámicamente el contenido de la página sin recargarla por completo.
- jQuery: jQuery es una biblioteca de JavaScript que simplifica la manipulación del DOM y la realización de solicitudes AJAX (Asynchronous JavaScript and XML), lo que la hace útil para implementar actualizaciones dinámicas en páginas web de manera sencilla.